Neural Word Embedding as Implicit Matrix Factorization

Shih-Ming Wang

NLPLab, Institute of Information Science, Academia Sinica

February 8, 2016

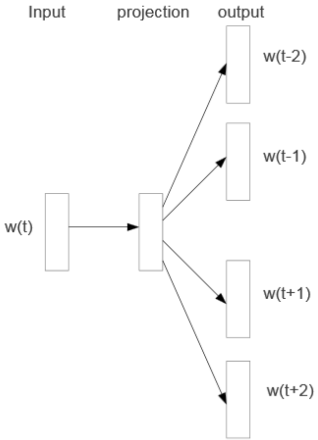
Outline

Introduction

- Word representation - why and how
- Distributional hypothesis
  - Words in similar contexts have similar meanings
  - Word-context matrix
- Skip-gram model
  - Neural-network based
  - Represent both word and context as vector
  - Maximize the dot-product of frequently occurring word-context vector pairs
  - Not well (theoretically) understood
- Analyze skip-gram as an implicit word-context matrix factorization

Skip-Gram Model

- Words as dense vector (projected from one-hot)
- Different projection matrix for input and output words (context)
- Loss function:
  $\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c} logp(w_{t+j}|w_t)$, where
  $p(w_o|w_t) = \frac{exp(\vec{w}_o\cdot\vec{w}_t)}{\sum_{i=1}^{||V||} exp(\vec{w}_i\cdot\vec{w}_t)}$
- Problem: softmax impractical because $||V||$ is usually large



Negative Sampling

- From now on, represent input and context word as w and c
- Skip-Gram optimizes conditional probability $P(c|w)$
- Negative Sampling optimizes joint probability $P(w, c)$
  - Maximize observed pairs $\sigma(\vec{w} \cdot \vec{c})$
  - Minimize k randomly sampled negative pairs $\sigma(\vec{w} \cdot \vec{c}_N)$
  - $\sigma(x) = \frac{1}{1+exp(-x)}$, Property: $\sigma(-x) = -\sigma(x)$
- Paired-wise loss function
  $L(w, c) = log(\sigma(\vec{w} \cdot \vec{c}) + k\mathbb{E}_{C_N \sim P(D)}\left[log(\sigma(-\vec{w} \cdot \vec{c}_N))\right]$
- Global loss function: $\sum_{w \in V_w} \sum_{c \in V_c} \#(w, c) L(w, c)$

SGNS as Implicit Matrix Factorization I

- SGNS learn a (input) word embedding W and a (output) context embedding C
- Rows of W are used as word vectors while C is usually ignored
- But effectively, SGNS is implicitly factorizing a word-context matrix $M = W \cdot C^T$
- Each cell of M equals $\vec{w} \cdot \vec{c}$, measuring the strength of association for a (w, c) pair
- But what is the explicit form $f(w, c) = \vec{w} \cdot \vec{c}$?

SGNS as Implicit Matrix Factorization II

- Assume that SGNS can fully reconstruct M (this requires infinite dimension for W, C)
- The global loss then equals the summation of M's cells $\sum_{w \in V_w} \sum_{c \in V_c} \#(w,c) \cdot L(w,c) = \sum_i \sum_j M_{ij} = \sum_i \sum_j f(w_i, c_j)$
- The global loss is optimized when each cell of M is optimized
- Goal: find explicit form $f(w,c) = \vec{w} \cdot \vec{c}$
- Strategy: Find $\sum_{w \in V_w} \sum_{c \in V_c} \ell(w,c) = \sum_{w \in V_w} \sum_{c \in V_c} \#(w,c) \cdot L(w,c)$, and let $\frac{\partial \ell(w,c)}{\partial(\vec{w} \cdot \vec{c})} = 0$ to find $\vec{w} \cdot \vec{c}$ that optimizes $\ell(w,c)$

SGNS as Implicit Matrix Factorization III

- $\ell(w,c) = \#(w,c) log\sigma(\vec{w},\vec{c}) + k \cdot \#(w) \cdot \frac{\#(c)}{|D|} log\sigma(-\vec{w} \cdot \vec{c})$
- $\vec{w} \cdot \vec{c} = log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c)}\right) - log\ k$ (Wait, PMI?)
- $M_{ij}^{SGNS} = W_i \cdot C_j = \vec{w_i} \cdot \vec{c_j} = PMI(w_i, c_j) - log\ k = M_{ij}^{PMI} - log\ k$
- M^{SGNS} was assumed to be fully reconstructed (not possible)
- In $\ell(w,c)$, the deviation of $\vec{w_i} \cdot \vec{c_j}$ from $M^{PMI} - log\ k$ is weighted by $\#(w,c)$ and $k \cdot \#(w) \cdot \frac{\#(c)}{|D|}$
- SGNS is a weighted matrix factorization of $M^{PMI} - log\ k$
- Deviation of frequent word-context pairs pay more

Point-wise Mutual Information

- $PMI(x,y) = log\frac{P(x,y)}{P(x)P(y)}$
- Empirically, $PMI(w,c) = log\frac{\#(w,c)\cdot|D|}{\#(w)\cdot\#(c)}$
- Issue: M^{PMI} is dense
  - For unobserved (w,c), $PMI(w,c) = log 0 = -\infty$
  - Solution: $PPMI(w,c) = max(PMI(w,c),0)$
  - *PPMI* lose information of infrequent word-context pairs
  - Intuition: prefer positive ('Canada','Snow') association against negative ('Canada','Desert') association
- $M^{SGNS} = M^{PMI} - log\ k$ is dense
- Modify: M^{SPPMI} , where $SPPMI(w,c) = max(PMI(w,c) - log\ k, 0)$

Singular Value Decomposition

- M^{SPPMI} could directly be used as word embedding
- But it's sparse, dense vectors sometimes could be better?
- $M^{SPPMI} = U \cdot \Sigma \cdot V^T \Rightarrow U_d \cdot \Sigma_d \cdot V_d^T = M^{SVD}$
- Choose $W^{SVD} = U_d \cdot \Sigma_d$ and $C^{SVD} = V_d$
  - Under-perform SGNS empirically
  - C^{SVD} is orthogonal while W^{SVD} is not
  - Both C^{SGNS} and W^{SGNS} are not orthogonal
  - $W^{SVD_{1/2}} = U_d \cdot \sqrt{\Sigma_d}$ and $C^{SVD_{1/2}} = V_d \cdot \sqrt{\Sigma_d}$
  - Generally $W^{SVD_\alpha} = U_d \cdot \Sigma^\alpha$(set to 1/2 in experiments)

Experiment Setting

- Corpus: English Wikipedia, 77.5 million sentences, 1.5 billion tokens
- Window size is set to 5, dropping words appears less than 100 times
- 189533 terms derived for both words and contexts

Deviation from Optimal

Method	PMI$-\log k$	SPPMI	SVD			SGNS		
			$d=100$	$d=500$	$d=1000$	$d=100$	$d=500$	$d=1000$
$k=1$	0%	0.00009%	26.1%	25.2%	24.2%	31.4%	29.4%	7.40%
$k=5$	0%	0.00004%	95.8%	95.1%	94.9%	39.3%	36.0%	7.13%
$k=15$	0%	0.00002%	266%	266%	265%	7.80%	6.37%	5.97%

- The optimal solution is $PMI - log\ k$
- SPPMI is near-perfect approximation of the optimum
- SVD is better when $d < 500$ and $k = 1$
- SVD fails to leverage higher dimension as SGNS does
- k's meaning for SGNS
  - higher k means more sample and better estimation of negative sample distribution (good news)
  - higher k means that negative examples are more probable (actually not a good thing)
- $SPPMI(w, c) = max(PMI(w, c) - log\ k, 0)$, only losing information as k increases

Linguistic Tasks

WS353 (WordSim) [13]			MEN (WordSim) [4]			Mixed Analogies [20]			Synt. Analogies [22]		
Representation		Corr.	Representation		Corr.	Representation		Acc.	Representation		Acc.
SVD	(k=5)	0.691	SVD	(k=1)	0.735	SPPMI	(k=1)	0.655	SGNS	(k=15)	0.627
SPPMI	(k=15)	0.687	SVD	(k=5)	0.734	SPPMI	(k=5)	0.644	SGNS	(k=5)	0.619
SPPMI	(k=5)	0.670	SPPMI	(k=5)	0.721	SGNS	(k=15)	0.619	SGNS	(k=1)	0.59
SGNS	(k=15)	0.666	SPPMI	(k=15)	0.719	SGNS	(k=5)	0.616	SPPMI	(k=5)	0.466
SVD	(k=15)	0.661	SGNS	(k=15)	0.716	SPPMI	(k=15)	0.571	SVD	(k=1)	0.448
SVD	(k=1)	0.652	SGNS	(k=5)	0.708	SVD	(k=1)	0.567	SPPMI	(k=1)	0.445
SGNS	(k=5)	0.644	SVD	(k=15)	0.694	SGNS	(k=1)	0.540	SPPMI	(k=15)	0.353
SGNS	(k=1)	0.633	SGNS	(k=1)	0.690	SVD	(k=5)	0.472	SVD	(k=5)	0.337
SPPMI	(k=1)	0.605	SPPMI	(k=1)	0.688	SVD	(k=15)	0.341	SVD	(k=15)	0.208

- Similarity test (WD353, Men)
  - SVD $\geq$ SPPMI $\geq$ SGNS (but the difference is small)
  - k matters, SVD, SPPMI prefer small k, SGNS prefers large k
- Analogies test (Mixed, Synt)
  - SVD is not good
  - SGNS significantly outperforms others on syntactic dataset
  - Might be due to that SGNS's higher weight for frequent word such as "the", "will", "each", "had"

Conclusion

- SGNS is shown to implicitly factorizing the $M^{PMI} - log\ k$ matrix
- M^{SPPMI}, where$SPPMI = max(PMI - log\ k, 0)$
  - Practical modification of SPMI
  - Approximate $M^{PMI} - log\ k$ better than SGNS
  - Not necessarily performs better on linguistic tasks
  - Might be due to SGNS's ability to perform weighted matrix factorization
- SVD
  - Leads to no significantly improvement but sometimes worse performance
  - The data is “big” enough to support M^{SPPMI}?